Option1\_GermanCredit

Daran Thach

3/05/2021

In this project, we describe some data from observations of 1000 applicants for credit worthiness in Germany. 30 variables are given, with ratings of “good credit” or “bad credit.” The predictor variables will be evaluated for their ability to predict good or bad credit risk. We analyze which of the variables are the best predictors of “good credit.”

knitr::opts\_chunk$set(echo = TRUE)  
  
# exploratory functions. Proceed with five to describe the data.  
german.df <- read.csv("GermanCredit.csv")  
# extract number of predicted good/bad cases  
length(german.df$RESPONSE)

## [1] 1000

# check to see if n/a values exist in predicted cases  
sum(!is.na(german.df$RESPONSE))

## [1] 1000

# all are integer values which are read. Check the outputs and type of data for each variable category  
str(german.df)

## 'data.frame': 1000 obs. of 32 variables:  
## $ OBS. : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ CHK\_ACCT : int 0 1 3 0 0 3 3 1 3 1 ...  
## $ DURATION : int 6 48 12 42 24 36 24 36 12 30 ...  
## $ HISTORY : int 4 2 4 2 3 2 2 2 2 4 ...  
## $ NEW\_CAR : int 0 0 0 0 1 0 0 0 0 1 ...  
## $ USED\_CAR : int 0 0 0 0 0 0 0 1 0 0 ...  
## $ FURNITURE : int 0 0 0 1 0 0 1 0 0 0 ...  
## $ RADIO.TV : int 1 1 0 0 0 0 0 0 1 0 ...  
## $ EDUCATION : int 0 0 1 0 0 1 0 0 0 0 ...  
## $ RETRAINING : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ AMOUNT : int 1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...  
## $ SAV\_ACCT : int 4 0 0 0 0 4 2 0 3 0 ...  
## $ EMPLOYMENT : int 4 2 3 3 2 2 4 2 3 0 ...  
## $ INSTALL\_RATE : int 4 2 2 2 3 2 3 2 2 4 ...  
## $ MALE\_DIV : int 0 0 0 0 0 0 0 0 1 0 ...  
## $ MALE\_SINGLE : int 1 0 1 1 1 1 1 1 0 0 ...  
## $ MALE\_MAR\_or\_WID : int 0 0 0 0 0 0 0 0 0 1 ...  
## $ CO.APPLICANT : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ GUARANTOR : int 0 0 0 1 0 0 0 0 0 0 ...  
## $ PRESENT\_RESIDENT: int 4 2 3 4 4 4 4 2 4 2 ...  
## $ REAL\_ESTATE : int 1 1 1 0 0 0 0 0 1 0 ...  
## $ PROP\_UNKN\_NONE : int 0 0 0 0 1 1 0 0 0 0 ...  
## $ AGE : int 67 22 49 45 53 35 53 35 61 28 ...  
## $ OTHER\_INSTALL : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ RENT : int 0 0 0 0 0 0 0 1 0 0 ...  
## $ OWN\_RES : int 1 1 1 0 0 0 1 0 1 1 ...  
## $ NUM\_CREDITS : int 2 1 1 1 2 1 1 1 1 2 ...  
## $ JOB : int 2 2 1 2 2 1 2 3 1 3 ...  
## $ NUM\_DEPENDENTS : int 1 1 2 2 2 2 1 1 1 1 ...  
## $ TELEPHONE : int 1 0 0 0 0 1 0 1 0 0 ...  
## $ FOREIGN : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ RESPONSE : int 1 0 1 1 0 1 1 1 1 0 ...

# check variable type for outcome variable 'RESPONSE.'  
class(german.df$RESPONSE)

## [1] "integer"

library(forecast)

## Warning: package 'forecast' was built under R version 3.6.3

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

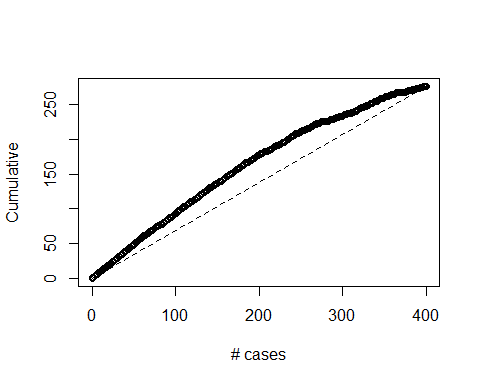
set.seed(2)  
  
# address overfitting issue by partitioning the data  
train.german <- sample(c(1:dim(german.df)[1]), dim(german.df)[1]\*0.6)  
train.df <- german.df[train.german, ]  
valid.df <- german.df[-train.german, ]  
  
#run logistic regression model, using glm w/family = binomial  
log.german <- glm(RESPONSE ~., data=train.df, family = binomial)  
options (scipen=999)  
summary (log.german)

##   
## Call:  
## glm(formula = RESPONSE ~ ., family = binomial, data = train.df)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.7622 -0.5973 0.3645 0.7077 2.3037   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.42534139 1.16113318 1.228 0.219618   
## OBS. -0.00045710 0.00039292 -1.163 0.244689   
## CHK\_ACCT 0.54899886 0.09662732 5.682 0.0000000133 \*\*\*  
## DURATION -0.04698932 0.01267424 -3.707 0.000209 \*\*\*  
## HISTORY 0.48967553 0.13201661 3.709 0.000208 \*\*\*  
## NEW\_CAR -1.11241122 0.49737010 -2.237 0.025313 \*   
## USED\_CAR 0.83401959 0.65714676 1.269 0.204387   
## FURNITURE -0.57184225 0.52245671 -1.095 0.273725   
## RADIO.TV -0.43375744 0.50086452 -0.866 0.386481   
## EDUCATION -1.25767574 0.68782210 -1.828 0.067476 .   
## RETRAINING -0.56706000 0.56175370 -1.009 0.312761   
## AMOUNT -0.00005204 0.00005991 -0.869 0.385028   
## SAV\_ACCT 0.24509062 0.07977444 3.072 0.002124 \*\*   
## EMPLOYMENT 0.22246316 0.10137398 2.194 0.028201 \*   
## INSTALL\_RATE -0.23002085 0.11367159 -2.024 0.043016 \*   
## MALE\_DIV -0.49785779 0.50903224 -0.978 0.328051   
## MALE\_SINGLE 0.32829423 0.27181740 1.208 0.227134   
## MALE\_MAR\_or\_WID -0.04895527 0.39757852 -0.123 0.902001   
## CO.APPLICANT 0.03381923 0.57209988 0.059 0.952861   
## GUARANTOR 1.73287372 0.61138098 2.834 0.004592 \*\*   
## PRESENT\_RESIDENT -0.01013088 0.11502911 -0.088 0.929819   
## REAL\_ESTATE 0.24809780 0.28224164 0.879 0.379387   
## PROP\_UNKN\_NONE -0.55032541 0.45656522 -1.205 0.228064   
## AGE 0.01289624 0.01150771 1.121 0.262432   
## OTHER\_INSTALL -0.45873647 0.27580111 -1.663 0.096255 .   
## RENT -0.11635493 0.59035464 -0.197 0.843755   
## OWN\_RES 0.10330124 0.55510480 0.186 0.852372   
## NUM\_CREDITS -0.54980261 0.23080463 -2.382 0.017214 \*   
## JOB -0.20643236 0.18983759 -1.087 0.276853   
## NUM\_DEPENDENTS -0.19522403 0.35229561 -0.554 0.579477   
## TELEPHONE 0.41910032 0.25896815 1.618 0.105588   
## FOREIGN 1.08235676 0.64658886 1.674 0.094141 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 726.13 on 599 degrees of freedom  
## Residual deviance: 521.48 on 568 degrees of freedom  
## AIC: 585.48  
##   
## Number of Fisher Scoring iterations: 5

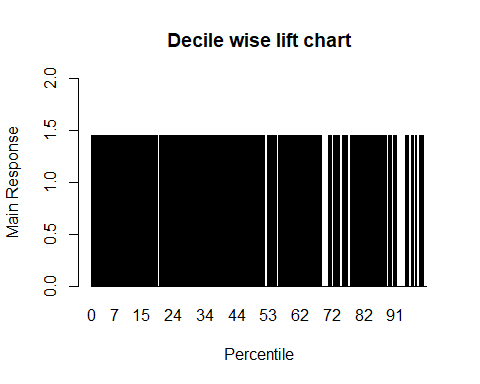
#evaluate classification performance; use predict () with type = response to  
#compute probabilities  
log.german.pred <- predict(log.german, valid.df, type="response")  
data.frame(actual = valid.df$RESPONSE[1:5], predicted = log.german.pred[1:5])

## actual predicted  
## 2 0 0.3533524  
## 6 1 0.7417094  
## 7 1 0.9431072  
## 11 0 0.4440879  
## 12 0 0.1071292

#evaluate classification performance; create lift, decile wise lift chart  
library(gains)  
gain <- gains(valid.df$RESPONSE, log.german.pred, groups=length(log.german.pred))  
#plot lift chart  
plot(c(0, gain$cume.pct.of.total\*sum(valid.df$RESPONSE)) ~ c(0, gain$cume.obs),  
 xlab ="# cases", ylab = "Cumulative", main ="")  
lines(c(0, sum(valid.df$RESPONSE)) ~ c(0, dim(valid.df)[1]), lty=2)



#compute deciles and plot decile-wise chart  
height <- gain$mean.resp/mean(valid.df$RESPONSE)  
midpt <- barplot(height, names.arg = gain$depth, ylim= c(0,2),   
 xlab= "Percentile", ylab= "Main Response",   
 main= "Decile wise lift chart")



# create confusion matrix to accurately find true positives and true negatives  
log.german.pred2 <- predict(log.german, valid.df)  
library(caret)

## Warning: package 'caret' was built under R version 3.6.3

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.3

tab.cf <- table(log.german.pred > 0.5, valid.df$RESPONSE)  
tab.cf

##   
## 0 1  
## FALSE 67 50  
## TRUE 57 226

# determine accuracy of confusion matrix (true positives/all positive & negative)  
sum(diag(tab.cf))/sum(tab.cf)

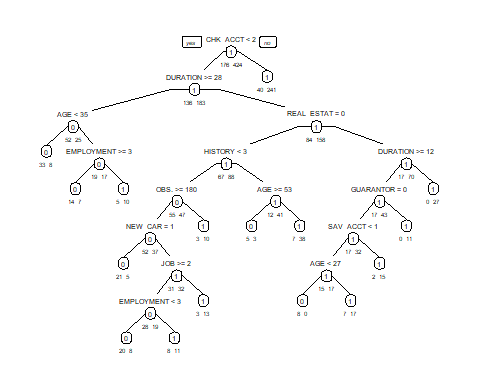
## [1] 0.7325

After we ran logistic regression and evaluated its performance, let’s run another classification predictor using CART (classification and regression trees).

# classification tree, using partitioned data above  
library(rpart)  
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.6.3

german.ct <- rpart(RESPONSE ~ ., data = train.df, method = "class")  
# plot classification tree  
prp(german.ct, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)



# run the confusion matrix for the above ct  
## run the matrix for training, then validation data  
## set argument type = 'class' to generate predicted class membership  
german.ct.pred.train <- predict(german.ct, train.df, type = "class")  
german.ct.pred.valid <- predict(german.ct, valid.df, type = "class")  
  
# default tree using training  
library(caret)  
# confusionMatrix(german.ct.pred.train, train.df$RESPONSE) gives error: 'data'  
## and 'reference' should be factors w/same levels  
## use table function to generate matrix  
tab.cm <- table(german.ct.pred.train, train.df$RESPONSE)  
tab.cm

##   
## german.ct.pred.train 0 1  
## 0 101 31  
## 1 75 393

# determine accuracy of training confusion matrix (true positives/all positive & negative)  
sum(diag(tab.cm))/sum(tab.cm)

## [1] 0.8233333

# Accuracy of training CART = 0.823  
# repeat matrix for validation data  
tab.cmv <- table(german.ct.pred.valid, valid.df$RESPONSE)  
tab.cmv

##   
## german.ct.pred.valid 0 1  
## 0 58 56  
## 1 66 220

# determine accuracy of validation confusion matrix (TP/TP+FP)  
sum(diag(tab.cmv))/sum(tab.cmv)

## [1] 0.695

# Accuracy of validation CART = 0.695  
Sys.time()

## [1] "2021-03-04 00:09:11 MST"

END OF RCODE using Rstudio.

Note: The Turnitin score >30% reflects a score due to necessary R code.

Learning discussion

The above assessment of *GermanCredit.csv,* was an in-depth R-based data mining of independent variables affecting the creditworthiness of a sample of German citizens. Our predictor variables ranged from presence of checking account, duration of credit, credit history, purposes of credit, employment status, et cetera. We will digest these variables to well known statistical analyses – logistic regression and classification/regression trees (CART)– and apply our final model after performance evaluation. Upon successful evaluation, we consider deployment of our prediction model.

We determined the data modeling task – classifying good or bad credit with the binary RESPONSE variable. After we explored the *GermanCredit.csv* data using several functions, we partitioned data 60:40 training to validation. We applied classification models using the data mining techniques of logistic regression and classification trees.

For the logistic regression model use a cutoff of 0.5 to predict probability of success (in this model, success = 1). Our results gave three statistically significant predictor variables – presence of checking account, duration of credit, and categorical credit history. Upon performing the model from each technique, we report the confusion matrix and the lift and decile-wise charts for the validation data. Here, our logistic regression model yielded just over 73%, while the CART method yielded 69%.

As our dataset contains many classifiers, we judge classification performance to search for the higher performing variables. Applying the lift chart shows the most accurate performance of the validation data beats the baseline model. We may apply our numbers of true good applicants to bad credit applicants in an opportunity cost table, where decisions can ultimately provide lines of credit from the net profit. Consequently, if our logistic regression model is applied towards future credit applicants, we would consider altering the probability of success cutoff threshold (>/=0.5) in extending lines of credit for better applicants. Therefore, our prediction model gave us a solid, customizable scoring mechanism for determining applicant line of credit.